ATARI®

ATARI

Microsoft BASIC II

Manuale Operativo

La Divisione Home Computer della Atari Inc ha dedicato molta attenzione alla preparazione ed alla stesura della documentazione del presente manuale e ritiene che le informazioni in esso contenute siano accurate ed attendibili.
Tuttavia la Atari declina ogni responsabilità, diretta o indiretta, imputabile ad errori ed omissioni e, poiche la Atari migliora ed aggiorna costantemente il software e l'hardware, non puo garantire la corrispondenza del prodotto a documentazione stampata dopo la presente data di pubblicazione.




A Warner Communications Company

Indice

Comandi

Comando	Esempi	Breve descrizione
AUTO	AUTO AUTO 10,10	Numera automaticamente le righe del programma. Se viene usato solo il comando, come nel primo esempio, il primo numero di riga à 100 e l'incremento tra i numeri di riga à 10. Nel secondo esempio il primo numero di riga è 10, il secondo 20 e così via.
CLOAD	CLOAD	Carica un programma codificato BASIC da una cassetta in memoria.
CONT	CONT	Prosegue l'esecuzione del programma dopo l'utilizzo dei comandi BREAK o STOP.
CSAVE	CSAVE	Memorizza un programma contenuto in memoria su di una cassetta in formato codificato.
DEL	DEL 10 DEL 10-100 DEL 10- DEL -10	Cancella una riga o una serie di righe da un programma. Per delimitare l'intervallo di righe occorre usare un trattino (-).
DOS	DOS	Richiama il menu del Sistema Operativo a Dischi (DOS). Per tornare al Microsoft BASIC II, si deve selezionare l'Opzione "B".
KILL	KILL "D:PROG1.AMB"	Cancella un programma da una determinata unità. Nell'esempio, il file "PROG1.AMB" viene cancellato dal dischetto inserito nell'unità a dischi.
LIST	LIST LIST 10 LIST 10-100 LIST 10- LIST -10 LIST "P:" LIST "C:"	Lista sullo schermo televisivo una serie di o tutte le righe del programma. Per delimitare l'intervallo di righe occorre usare un tratino ( ). Il suddetto elenco può essere eseguito anche sulla stampante, su cassetta o su un'altra unità specifica.

Comandi

Comando	Esempi	Breve descrizione
LOAD	LOAD "D:EXAMPLE1.AMB" LOAD "C:"	Carica un programma sulla memoria del calcolatore. Tale programma può essere su discchetto o su cassetta.
LOCK	LOCK "D:PROGRAM2.AMB"	Protegge il programma che si trova sul dischetto da cancellazioni accidentali.
MERGE	MERGE "D:EXAMPLE2.AMB" MERGE "C:"	Accoda un programma che si trova su dischetto o cassetta ad uno già residente in memoria. Tutti i numeri di riga doppi presenti nel file contenuto sul dischetto sostituiscono quelli del file presente in memoria.
NAME...TO	NAME "D:OLDFILE" TO "NEWFILE"	Modifica il nome di un programma su un'unità. Nell'esempio, il file di nome "OLDFILE" viene rinominato con nome "NEWFILE".
NEW	NEW	Cancella un programma dalla memoria.
RENUM	RENUM RENUM 100,50,10	Rinumera le righe del programma. Se non vengono usati parametri, come nel primo esempio, il numero della prima riga del programma viene posizionato a 10 e le restanti righe vengono incrementate di 10 in 10. Nel secondo esempio, la riga 50 del programma viene rinumerata a 100 e le restanti righe vengono incrementate con passo 10.
RUN	RUN RUN 100	Inizia l'esecuzione del programma. Nel secondo esempio l'esecuzione del programma inizia a partire dalla riga 100.
SAVE	SAVE "D:TESTFILE.AMB"	Salva un programma in formato simbolico (tokenized) su un'unità.
SAVE...LOCK	SAVE "D:TESTFILE.AMB"LOCK	Salva un programma su dischetto in formato codificato e blocca il file per prevenirne la distruzione. Un programma bloccato non può essere nè stampatto nè modificato.
TROFF	TROFF	Disattiva il meccanismo di traccia

Comandi

Comando	Esempi	Breve descrizione
		mento per la ricerca di eventuali errori.
TRON	TRON	Attiva il meccanismo di tracciamento per la ricerca di eventuali errori.
UNLOCK	UNLOCK "D:TESTFILE.AMB"	Sblocca un file su dischetto in modo da poterci scrivere, poterlo cancellare o potrlo rinominare.
VERIFY	VERIFY "D:FILEPROG.AMB" VERIFY "C:"	Confronta due programmi, uno residente in memoria e l'altro su dischetto o cassetta. Se essi non corrispondono esattamente viene emesso il messaggio di errore "TYPE MISMATCH ERROR" (Errore di mancata corrispondenza).

Note

Statements

Statements	Esempi	Breve descrizione
AFTER	AFTER(3600) 125	Imposta il contatore temporale esprimendo il tempo in sessantesimi di secondo. Nell'esempio, dopo 3.600 sessantesimi (= 1 minuto) il programma prosegue dalla riga 125.
CLEAR	CLEAR	Azzera tutte le variabili annulla tutte le stringhe e tutte le matrici (arrays) del programma.
CLEAR STACK	CLEAR STACK	Annulla tutti i contatori temporali. Può essere usato per causare l'aborto dello statement AFTER.
CLOSE	CLOSE#1	Chiude un file precedentemente aperto. Il segno ed il numero che lo segue sono obbligatori per identificare il blocco di controllo di input/output.
COMMON	COMMON ALL COMMON L,L1(2),L$	Mantiene intatte le variabili di programma. Con questo statement si può mantenere il contenuto di una, di molte o di tutte le variabili nel passaggio da un programma ad un altro.
DEF	DEF AVG(X,Y)=(X+Y)/2	Permette all'utente di definire le proprie funzioni. Si possono definire sia funzioni di stringa che funzioni numeriche. Le funzioni di stringa definite dall'utente sono disponibili solo quando si usa il dischetto di ampliamento.
DIM	DIM A$(35) DIM NUM(10,5,2)	Definisce il numero di elementi di una matrice numerica o di una stringa. Una matrice può essere multidimensionale.
END	END	Termina l'esecuzione di un programma. E' l'ultimo statement utilizzato in un programma. Esso chiude tutti i files ed azzera tutti i contatori temporali.

Statements

Statements	Esempi	Breve descrizione
ERROR	ERROR 2	Forza la valutazione di un errore in un programma (debugging) per verificare il comportamento del programma in caso di errore. Si possono impostare sia errori BASIC che di sistema.
FOR...TO...STEP /NEXT	FOR X=0 TO 10	Imposta un contatore per l'esecuzione ripetuta di uno o di un gruppo di statements. Esegue tutti gli statements presenti prima del comando NEXT, finchè il contatore raggiunge il numero specificato dopo TO. Se viene usato STEP, il contatore viene incrementato del valore specificato da STEP.
GET	GET#1,X GET#1,AT(8,2)	GET e PUT sono statements opposti l'uno all'altro. GET legge un singolo byte e lo memorizza nella variabile specificata dopo la virgola.
GOSUB /RETURN	GOSUB 100	Provoca un salto incondizionato da una riga di programma con ritorno al successivo statement. Viene usato per richiamare una subroutine.
GOTO	GOTO 50	Opera un salto incondizionato ad una riga per continuare l'esecuzione.
IF...THEN	IF X=1 THEN Y=X	Verifica equazioni e disequazioni tra stringhe e numeri per stabilire se la condizione è vera o falsa. Se la condizione è vera il programma esegue il comando dopo lo statement THEN. In caso contrario, continua l'esecuzione a partire dalla riga successiva.
IF...THEN...ELSE	IF X=0 THEN Y<>X ELSE Y=X	E' uguale a "IF...THEN" con l'eccezione che, per una condizione falsa, l'esecuzione del programma passa al comando che segue lo statement ELSE.

Statements

Statements	Esempi	Breve descrizione
INPUT	INPUT I INPUT I$ INPUT "Type a number";I INPUT "Your name?";I$	Arresta l'esecuzione di un programma per accettare informazioni da un'altra unità (l'unità standard è la tastiera). Lo statement INPUT permette di assegnare valori a variabili numeriche e a stringhe. Un punto interrogativo viene usato come richiesta di inserimento dati, a meno che non venga usata la stringa opzionale di richiesta.
INPUT_AT	INPUT AT (2,4)I$	E' uguale a INPUT con l'eccezione che il programma accetta l'introduzione di un dato in una specifica posizione dello schermo televisivo. (Nell'esempio, colonna 2, riga 4).
LET	LET Z=2	E' uno statement opzionale per l'assegnazione di variabili. E' anche accettabile Z=2.
LINE INPUT	LINE INPUT "Name:";I$	E' uguale allo statement INPUT con l'eccezione che un'intera riga (compresi spazi, virgole, punti e virgole ed altri separatori) può essere introdotta da tastiera o dall'unità specificata. Un "?" viene usato come richiesta di inserimento dati, a meno che non venga usata la stringa opzionale di richiesta.
LINE_INPUT AT	LINE INPUT AT (4,4)I LINE INPUT AT (6,8)"Nome.",I$	E' uguale a LINE INPUT con l'eccezione che il programma accetta l'input a partire da una determinata posizione dello schermo televisivo (colonna, riga).
MOVE	MOVE 55,222,5	Muove zone di memoria da una locazione all'altra. Nell'esempio i primi cinque bytes di memoria a partire dall'indirizzo 55 (decimale) vengono trasferiti a partire dall'indirizzo 255. Si possono usare anche numeri esadecimali per specificare gli indirizzi.

Statements

Statements	Esempi	Breve descrizione
NEXT	NEXT NEXT I	Termina lo statement FOR...TO..STEP. Il nome della variabile è opzionale (vedere FOR...TO...STEP/NEXT).
NOTE	NOTE #4,I,J	Localizza il successivo byte che deve essere letto su un file su dischetto. Nell'esempio, NOTE memorizza la posizione del numero di settore corrente in J.
ON ERROR	ON ERROR 550	Provoca l'esecuzione di un programma a partire da una determinata riga quando si incontra un errore. Nell'esempio, il programma, se si verifica un errore, prosegue dalla riga 550. RESUME è richiesto per riportare l'esecuzione alla routine originale.
ON...GOSUB /RETURN	ON G GOSUB 100,200,300	Stabilisce quale subroutine verrà eseguita successivamente. Nell'esempio, a seconda che G assuma i valori 1,2 o 3, il programma salterà rispettivamente alla riga 100,200 o 300 (vedi RETURN).
ON...GOTO	ON G GOTO 100,200,300	Stabilisce quale subroutine verrà eseguita successivamente. Nell'esempio fornito,a seconda che G assuma i valori 1,2 o 3, l'esecuzione del programma salterà alla riga 100, 200 o 300.
OPEN	OPEN #3,"P:" OUTPUT OPEN #4,"D:PROGRAM.AMB" INPUT	Apre un file per permettere le operazioni di lettura o scrittura. Lo statement identifica il blocco di controllo di input/output usato da una specifica unità, per esempio una stampante o un'unità dischi, ed indica il tipo di operazione da eseguire (UPDATE, APPEND, INPUT o OUTPUT).
OPTION BASE	OPTION BASE 1 [illegible]	Indica il valore iniziale dell'indice di tutte le matrici e di tutte le variabili dimensionate. Permette all'utente di stabilire il valore

Statements

Statements	Esempi	Breve descrizione
		iniziale dell'indice usato nei loops e nelle matrici. Il valore standard è zero (0). L'esempio fornito stabilisce che tutti gli indici del programma inizieranno automaticamente da 1.
OPTION CHR	OPTION CHR1 OPTION CHR2 OPTION CHR0	Riserva bytes di memoria per dati di caratteri. OPTION CHR1 riserva 1024 bytes, OPTION CHR2 riserva 512 bytes e OPTION CHR0 rilascia tutte le zone di memoria riservate da OPTION CHR (vedere VARPTR).
OPTION PLM	OPTION PLM1 OPTION PLM2 OPTION PLM0	Riserva Bytes di memoria per la grafica del giocatore-missile. OPTION PLM1 riserva 1024 bytes, OPTION PLM2 riserva 512 bytes e OPTION PLM0 rilascia tutte le zone di memoria riservate da OPTION CHR (vedere VARPTR).
OPTION RESERVE	OPTION RESERVE 24	Riserva bytes di memoria per routines in linguaggio macchina. Vedere VARPTR.
PRINT	PRINT "HELLO" PRINT 25*4 PRINT "Reply=";R$?Y	Visualizza variabili, costanti numeriche e stringhe sullo schermo televisivo. Usato da solo lo statement PRINT esegue la stampa di una riga bianca sullo schermo TV. Al posto della parola PRINT può essere usata anche il simbolo di punto interrogativo (?).
PRINT_AT	PRINT AT(4,4)RX$	Stampa variabili, costanti numeriche e stringhe in una determinata posizione (colonna e riga) dello schermo TV o un determinato settore di un file su dischetto.
PRINT_SPC	PRINT SPC(5)"Hi!";SPC(5)"Bye"	Stampa il numero di spazi indicati tra parentesi a partire dall'attuale posizione del cursore. Differisce da TAB, poichè TAB conta

Statements

Statements	Esempi	Breve descrizione
		gli spazi a partire dalla colonna più a sinistra.
PRINT_TAB	PRINT TAB(5)"Hello"	Stampa il numero di spazi indicati tra parentesi iniziando dalla colonna più a sinistra del campo di testo.
PRINT USING	PRINT USING"# ",1	Permette all'untente di formattare il suo testo in 12 modi:
	PRINT USING"##",NUMBER	- allinea i numeri nelle colonne indicate dal simbolo .
	PRINT USING"###.##";MONEY	- posiziona un punto decimale nel risultato.
	PRINT USING"###,###.##";AMT#	- suddivide il numero a gruppi di tre cifre (migliaia) con una virgola.
	PRINT USING"** ###.##";CASH	- riempie gli spazi bianchi con asterischi.
	PRINT USING "$ ###.##";DOLLAR	- stampa un segno di dollaro ($) prima della prima cifra a sinistra.
	PRINT USING "$$###.##",CHECK	- stampa un segno di dollaro mobile ($) davanti al risultato.
	PRINT USING "**$###.##",FLOAT	- combina $ mobile e campi di * nel risultato.
	PRINT USING "##^^^^";EXPONENT	stampa il risultato in formato esponenziale (E o D).
	PRINT USING"+###";PLUS	stampa un segno più (+) prima o dopo il risultato.
	PRINT USING"### ";MINUS	- stampa un segno meno (-) prima o dopo il risultato.
	PRINT USING"!",INITIAL$	- estrae il primo carattere di una stringa.
	PRINT USING"%%",PART$	estrae una parte di una stringa.
PUT	PUT #6, ASC("A"),	PUT e GET sono statemens opposti. PUT scrive un singolo byte (0 255) su uno specifico file o unità.
RANDOMIZE	RANDOMIZE RANDOMIZE 20	Cambia seme alla funzione RND per assicurare che si verifichi una diversa sequenza di numeri casuali ogni volta che viene eseguito un programma. Il secondo esempio assicu

Statements

Statements	Esempi	Breve descrizione
		ra che una sequenza casuale venga generata ripetutamente.
READ DATA	READ A,B,C DATA 1,2,3	Assegna numeri o stringhe nello statement DATA a variabili presenti nello statement READ.
REM ola'	REM Ignora questo commento ! Ignora questa nota ' Ignora anche questa nota X=1;REM è necessario il punto e virgola X=1'non è necessario il punto e virgola	Permette di inserire commenti in un programma. Durante l'esecuzione del programma gli statements REM (abbreviazione di REMARK) vengono ignorati. Al posto della parola REM si può usare un punto esclamativo o un apostrofo.
RESTORE	RESTORE RESTORE 110	Permette di riutilizzare gli statements DATA. Se non viene usato alcun parametro, READ rilegge i dati dal primo statement DATA. Nel secondo esempio, RESTORE fa si che lo statement READ inizi a leggere dati dalla riga 110.
RESUME	RESUME RESUME 55 RESUME NEXT	Aiuta il ripristino del programma dopo un errore o un'interruzione causata da un contatore temporale. RESUME riattiva l'esecuzione del programma a partire dalla riga in cui si è verificato l'errore o l'interruzione. Se viene usato un numero di riga il programma riprende da tale riga. Se viene usato uno statement NEXT, l'esecuzione del programma riprende dallo statement successivo a quello in cui si è verificato l'errore o l'interruzione (può essere sulla stessa riga). RESUME completa gli statements ON ERROR e AFTER.
RETURN	RETURN	Completa gli statements GOSUB e ON...GOSUB e restituisce il controllo al programma dopo l'esecuzione di una subroutine (Vedi GOSUB).

Statements

Statements	Esempi	Breve descrizione
STACK	PRINT STACK IF STACK=0 THEN PRINT"Stack full"	Fornisce il numero d'introduzioni disponibili nella pila (stasck) dei tempi (Usata per memorizzaez i sessantesimi di secondo per AFTER e SOUND).
STOP	STOP	Arresta l'esecuzione del programma. Usare CONT per proseguire l'esecuzione del programma (iniziando dalla riga successiva).
WAIT...AND	WAIT £D40B,AND £FF,110	Arresta il programma in attesa che si verifichino deteminate condizioni. Tecniche avanzate usano WAIT per gestire VBLANK. Nell'esempio, il programma ricerca il contenuto dell'indirizzo £D40B, lo sottopone ad AND logico con £FF ed attende finchè non sia uguale a 110.

Funzioni

Funz. Numeriche	Esempi	Breve descrizione
ABS	Y=ABS(-7)	Calcola il valore assoluto di un numero.
ATN	X=ATN(5.3)	Calcola la funzione arcotangente trigonometrica di un dato numero.
COS	PRINT COS(.95)	Calcola la funzione trigonometrica coseno di un dato numero.
EXP	EULER=EXP(3)	Calcola il numero di Eulero (e) ele vato alla potenza del numero specifi cato tra parentesi.
INT	PERINT INT(5.3)	Restituisce l'intero di un numero arrotondandolo per difetto.
LOG	L=LOG(.5)	Calcola il logaritmo naturale di un numero positivo e diverso da zero.
RND	R=RND PRINT RND(0) NUMBER=RND(100)	Genera numeri casuali in semplice precisione. RND usato da solo o con uno zero tra parentesi, produce un valore casuale compreso tra zero e uno. Usato con un numero diverso da zero, produce un valore intero compreso tra uno e tale numero. Nel l'esempio, RND(100) genera un numero casuale compreso tra 1 e 100.
SGN	PRINT SGN(R*B)	Rilascia il segno di un numero. Se il numero è positivo, il valore re stituito è +1. Se il numero è zero, il valore restituito è zero. Se il numero è negativo, il valore resti tuito è -1.
SIN	PRINT SIN(1)	Calcola il valore della funzione trigonometrica seno per un dato numero.
SQR	ROOT=SQR(25)	Calcola la radice quadrata di un numero positivo.
TAN	PRINT TAN(.22)	Calcola il valore della funzione trigonometrica tangente per un de terminato numero.
+(Concatenazione)	PRINT INITIAL$+NAME$	Concatena due stringhe

Funzioni

Funz.Numeriche	Esempi	Breve descrizione
ASC	PRINT ASC("Simona")	Trasforma il primo carattere contenuto tra parentesi in codifica decimale ATASCII. Nell'esempio, il carattere è "S" che corrisponde a 83 (decimale) in codice ATASCII.
CHR$	A$=CHR$(65)	Converte il codice ATASCII contenuto tra parentesi in una stringa di un carattere. La funzione CHR$ è opposta alla funzione ASC.
INKEY$	R$=INKEY$	Restituisce codifica ultimo tasto.

Funz. Stringa	Esempi	Breve descrizione
INSTR	HOLD=INSTR(5,A$,"IL")	Ricerca una serie di caratteri contigui all'interno di una stringa e rilascia la posizione del primo carattere della serie cercata allo interno della stringa stessa. Se la sottostringa non viene trovata, restituisce il valore zero. Nell'esempio, viene ricercata la sottostringa "IL" a partire dal quinto carattere della stringa A$. Se non viene forni to nessun numero di inizio ricerca, la ricerca avrà inizio dal primo carattere della stringa principale.
LEFT$	PRINT LEFT$("LEFTY",4)	Partendo da sinistra, restituisce una sottostringa formata da tanti caratteri quanti specificati dal valore presente nell'istruzione. Nell'esempio, verrà stampato "LEFT".
LEN	PRINT LEN(C$)	Restituisce la lunghezza di una stringa (numero di caratteri).
MID$	PRINT MID$("THEMIDPART",4,3)	Estrae una serie di caratteri dalla parte centrale di una stringa. Nello esempio, l'estrazione di 3 caratteri nella stringa A$, a partire dal quarto carattere, provoca la stampa di "MID".

Funzioni

Funz. Stringa	Esempi	Breve descrizione
RIGHT$	A$=RIGHT$("THERIGHT",5)	Partendo da destra restituisce una sottostringa formata da tanti caratteri quanti specificati dal valore presene nell'istruzione. Nell'esempio, alla stringa A$ è assegnato "RIGHT".
SCRN$	C$=SCRN$(5,5) PRINT ASC(C$)	Restituisce il valore del carattere durante operazione in modo-testo. Nei modi grafici rilascia il numero del registro colore in codice AIASII (ad eccezione dei modi grafici 4 e 6).
STR$	X=STR$(99.99)	Converte un numero in una stringa. E' l'opposto della funzione VAL.
STRING$	PRINT STRING$(36,"*") PRINT STRING$(36,42)	Restituisce una stringa di caratteri. Nel primo esempio, vengono stampate 36 ripetizioni di una stringa (lo asterisco). Nel secondo esempio, vengono stampate 36 ripetizioni della funzione CHR$ (42 è il valore dell'asterisco in codice AIASII). In altre parole, il risultato di ambedue gli esempi è una stringa di 36 asterischi.
TIME$	PRINT TIME$	Restituisce il contenuto della stringa TIME$ contenenti i valori: ora, minuti e secondi (12:01:00).
VAL	PRINT VAL(H$)	Converte una stringa in un numero. E' l'opposto della funzione STR$.
EOF	EOF (4)	Restituisce il valore di vero o falso per indicare la condizione di fine-file dell'ultima lettura da un blocco di controllo di input-output.

Funzioni

Funz.Speciliz.	Esempi	Breve descrizione
ERL	PRINT ERL	Restituisce il numero di riga dello ultimo errore riscontrato.
ERR	PRINT ERR	Restituisce il numero di codice dell'ultimo errore riscontrato.
FRE(0)	PRINT FRE(0)	Fornisce il numero di bytes RAM liberi, disponibili per l'uso.
PEEK	PRINT PEEK(751) ADRS=PEEK(PLACE)	Fornisce il contenuto della memoria a partire dall'indirizzo racchiuso tra parentesi. L'indirizzo e il byte possono essere un numero o una variabile espressi in decimale o in esadecimale. Si possono ricercare indirizzi ROM o RAM.
POKE	POKE 1034,255 POKE ADRS,J POKE PLACE,X*Y/2	Inserisce un byte in una specificata posizione della memoria RAM (ma non ROM). L'indirizzo e il byte possono essere un numero, un'espressione aritmetica, o una variabile, espressi in decimale o esadecimale.
STATUS	PRINT STATUS(4)	Contiene il valore del quarto byte del blocco di controllo di input/output, il valore indica la condizione di errore (1=errore; 0=nessun errore).
TIME	PRINT TIME	Restituisce il contenuto del contatore di tempo (RTCLOCK). Diversamente da TIME$ (che fornisce il tempo trascorso in ore, minuti e secondi), TIME fornisce il valore in sessantesimi di secondo.
USR	X=USR(898,J)	Passa il controllo del programma ad una subroutine in linguaggio macchina. Nell'esempio, la funzione USR passa il contatore ad una subroutine all'indirizzo 898 decimale. Assieme all'indirizzo può essere abbinato un parametro opzionale, utilizzabile dalla subroutine.

Funzioni

Funz.Specializ.	Esempi	Breve descrizione
VARPTR	ADRS=VARPTR(A$) PRINT VARPTR(A$)+1 PLAYR1=VARPTR(PLM)1 PRINT VARPTR(CHR1) PRINT VARPTR(RESERVE)	Fornisce l'indirizzo di memoria delle entrate della tabella delle variabili. Nel primo esempio, VARPTR fornisce il numero di bytes della stringa. Nel secondo esempio, viene stampato l'indirizzo di inizio della stringa. Negli altri esempi, VARPTR fornisce rispettivamente l'indirizzo (MSB,LSB) del primo byte assegnato ai grafici missile-giocatore, un set di caratteri e la memoria riservata per i programmi in linguaggio assembler.

Caratteristiche dei Giochi

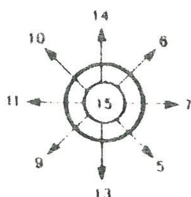
Grafici	Esempi	Breve descrizione
CLS	CLS CLS 25	Azzera le aree di testo dello schermo e posiziona il registro del colore di fondo sul colore indicato. Nel funzionamento a schermo pieno (Full sceen), il numero opzionale specificato dopo CLS stabilisce la luminosità ed il colore del bordo. Quando si utilizza lo schermo suddiviso, stabilisce la luminosità ed il colore del fondo.
COLOR	COLOR 4	Indica il registro di colore da usare per grafica a colori.
FILL	FILL 5,5 TO 5,20	Riempie un'area dello schermo televisivo con il colore corrispondente al numero di registro indicato.
GRAPHICS	GRAPHICS 0 GRAPHICS 2+16 GRAPHICS 7+32	Seleziona uno dei 12 modi grafici (CTIA fornisce 8 modi grafici). Aggiungendo +16 ad un modo grafico, l'istruzione fornisce una completa visualizzazione dello schermo. Aggiungendo +32, impedisce ai comandi grafici di azzerare lo schermo.
PLOT PLOT...TO	PLOTX,Y PLOT 5,5 TO 10,5	Disegna sullo schermo un punto, una riga o alcune righe continue.
SETCOLOR	SETCOLOR 5,4,10	Associa un colore ed una luminosità ad un registro di colore. Il primo parametro specifica il registro usato (0-3 è valido per i missili-giocatori, 4-7 per i colori del campo di gioco e 8 per il registro di fondo). Il secondo parametro rappresenta il numero del colore (0-15), mentre il terzo parametro rappresenta la luminosità (data da un numero pari compreso tra 0 e 14, più alto è il numero più brillante è la luminosità).

Caratteristiche dei Giochi

Grafici	Esempi	Breve descrizione
Paddle Controlless (2x4 gruppi)	PADDLE(0)=624 PADDLE(1)=625 PADDLE(2)=626 PADDLE(3)=627 PADDLE(4)=628 PADDLE(5)=629 PADDLE(6)=630 PADDLE(7)=631	Gli stati dei comandi a manopola vengono memorizzati agli indirizzi da 624 a 632. Per i comandi a manopola esistono quattro bocchettoni ed ognuno gestisce due governi. Lo stato della manopola più a sinistra è memorizzato all'indirizzo 624, e così via. L'intervallo di valori dello stato varia da 1 a 228 man mano che la manopola viene girata in senso antiorario.

Caratteristiche dei Giochi

Governi gioco	Numero	Locazione PEEK	Breve descrizione
Grilletto Manopola (2x4 gruppi)	PTRIG(0)	=636	I valori determinati dall'azionamento del grilletto (pulsante) associato alla manopola sono memorizzati agli indirizzi da 636 a 643. Il valore di sparo della manopola più a sinistra è memorizzato all'indirizzo 636 e così via. Se il grilletto è stato azionato, PEEK(PTRIG(n)) trova uno zero (0) all'indirizzo. Se il grilletto non è stato azionato, l'indirizzo conterrà un uno (1).
	PTRIG(1)	=637	
	PTRIG(2)	=638	
	PTRIG(3)	=639	
	PTRIG(4)	=640	
	PTRIG(5)	=641	
	PTRIG(6)	=642	
	PTRIG(7)	=643	
Governi Cloche	STICK(0)	=632	Gli stati dei comandi a cloche vengono memorizzati agli indirizzi 632 635. Lo stato con la cloche tutta a sinistra è memorizzato all'indirizzo 632 e così via. Ogni stato contiene uno dei valori mostrati in figura 1.
	STICK(1)	=633	
	STICK(2)	=634	
	STICK(3)	=635	
Grilletto Cloche	STRIG(0)	=644	I valori determinati dall'azionamento del grilletto (pulsante) associato alla cloche sono memorizzati agli indirizzi da 644 a 647. Il valore di sparo della cloche di sinistra è memorizzato all'indirizzo 644 e gli altri di seguito. Il byte di stato contiene uno zero (0) se il grilletto è stato azionato, ed un uno (1) in caso contrario.
	STRIG(1)	=645	
	STRIG(2)	=646	
	STRIG(3)	=647	
Tasti Console	OPTION	PEEK(53279)=3	Lo stato dei tasto OPTION, SELECT e START è memorizzato all'indirizzo 53279. Il byte di stato contiene il valore 7 finchè nessuno dei tre tasti è premuto. Poi assume i valori 3,5 o 6 a seconda del tasto utilizzato.
	SELECT	PEEK(53279)=5	
	START	PEEK(53279)=6	
	No Key	PEEK(53279)=7	



Codici di Errore

Per una descrizione più dettagliata dei seguenti errori, consultare il Manuale "Atari Microsoft BASIC II - Manuale di Riferimento - Appendice O".

Codice Errore	Spiegazione
1	NEXT senza FOR
2	Errore di sintassi
3	RETURN senza GOSUB
4	Fine dei dati in DATA (esauriti)
5	Errore di richiamo funzione
6	Superamento (Overflow)
7	Fine della memoria
8	Riga non definita
9	Indice fuori gamma
10	Erore di ridimensionamento
11	Divisione per zero
12	Uso diretto non valido
13	Mancata corrispondenza del tipo
14	Errore di I/O del file
15	Quantità troppo elevata
16	Formula troppo complessa
17	Non si può continuare
18	Funzione utente non definita
19	Nessun "RESUME"
20	RESUME senza errore
21	FOR senza NEXT
22	Errore "LOCK" (bloccaggio)
23	Errore di tempo
128	Interruzione con "BREAK"
129	Errore IOCB
130	Unità non esistente
131	Errore di sola scrittura IOCB
132	Comando non valido
133	Unità o file non aperto
134	Numero IOCB errato
135	Errore di lettura nell'Input/Output Control Black (IOCB)
136	EOF (End Of File)
137	Record troncato
138	Superamento tempo disponibile a una periferica.
139	NAK Unità indisponibile.
140	Bus seriale
141	Cursore fuori campo
142	Errore di sovraccarico di dati nel bus seriale
143	Errore del totale di controllo del pacchetto dati dal bus seriale
144	Errore generato nella unità
145	Errore di confronto di lettura dopo la scrittura
146	Funziona non implementata
147	Insufficiente memoria RAM
160	Numero di unità errato
161	Sono stati aperti troppi files.
162	Disco pieno
163	Errore di I/O dei dati di sistema, non recuperabile
164	Confronto numero di file errato
165	Errore nel nome del file
166	Errore di lunghezza dei dati POINT
167	File bloccato
168	Comando non valido
169	Directory del disco piena
170	File non trovato
171	POINT non valido

ATARI®


A Warner Communications Company

ATARI International (Italy) Inc.
Viale della Liberazione, 18
20124 MILANO